BAB II

TINJAUAN PUSTAKA

2.1 Sejarah kriptografi

Kriptografi memiliki sejarah yang panjang dan kaya, bermula dari kebutuhan manusia untuk menyembunyikan informasi penting dari pihak yang tidak berwenang. Sejarah kriptografi dapat ditelusuri kembali hingga ribuan tahun yang lalu, ketika peradaban kuno mulai mengembangkan metode untuk melindungi komunikasi rahasia mereka. Salah satu contoh tertua penggunaan kriptografi ditemukan dalam hieroglif Mesir kuno sekitar 1900 SM, di mana seorang juru tulis menggunakan simbol hieroglif yang tidak biasa untuk menyembunyikan makna sebenarnya dari tulisan di makam seorang bangsawan(Naser, n.d.).

Peradaban Yunani kuno juga memberikan kontribusi signifikan dalam perkembangan kriptografi dengan menciptakan alat yang dikenal sebagai "*scytale*". Alat ini berupa batang kayu dengan diameter tertentu yang digunakan oleh tentara Sparta untuk mengirim pesan rahasia. Pesan ditulis pada pita kulit yang dililitkan pada scytale, dan hanya penerima yang memiliki batang dengan diameter yang sama yang dapat membaca pesan tersebut dengan benar. Metode ini menunjukkan pemahaman awal tentang konsep kunci dalam kriptografi, di mana diameter batang berfungsi sebagai kunci untuk membuka pesan rahasia(Radanliev, 2023).

Era Romawi kuno menandai perkembangan penting dalam sejarah kriptografi dengan diperkenalkannya Caesar Cipher oleh Julius Caesar sekitar 50 SM. Cipher ini menggunakan metode substitusi sederhana di mana setiap huruf dalam alfabet digeser dengan jumlah posisi tertentu. Meskipun sederhana menurut standar modern, Caesar Cipher menjadi fondasi bagi pengembangan sistem kriptografi substitusi yang lebih kompleks. Keberhasilan Caesar dalam menggunakan cipher ini untuk komunikasi militer menunjukkan efektivitas kriptografi dalam konteks strategis dan militer(Kumar Sharma et al., 2021).

Abad pertengahan menyaksikan perkembangan kriptografi yang lebih sophisticated dengan kontribusi dari dunia Islam. Pada abad ke-9, Al-Kindi,

seorang polymath Arab, menulis risalah berjudul "A Manuscript on Deciphering Cryptographic Messages" yang dianggap sebagai karya pertama tentang cryptanalysis. Al-Kindi memperkenalkan teknik frequency analysis untuk memecahkan cipher substitusi, sebuah metode yang tetap relevan hingga saat ini. Kontribusi ini menandai dimulainya era cryptanalysis sebagai disiplin ilmu yang sistematis dan menunjukkan bahwa keamanan kriptografi tidak hanya bergantung pada kerahasiaan metode, tetapi juga pada kekuatan matematis dari algoritma yang digunakan(De Leeuw et al., 2007).

Breakthrough paling signifikan dalam sejarah kriptografi modern terjadi pada tahun 1976 dengan publikasi paper "New Directions in Cryptography" oleh Whitfield Diffie dan Martin Hellman. Paper ini memperkenalkan konsep revolutionary tentang public-key cryptography atau asymmetric cryptography, yang memecahkan masalah fundamental dalam kriptografi klasik yaitu key distribution problem. Konsep ini kemudian diimplementasikan dalam algoritma RSA oleh Ron Rivest, Adi Shamir, dan Leonard Adleman pada tahun 1977, yang menjadi foundation bagi secure communication dalam era internet modern(Madhavan & Saxena, 2003).

Era kontemporer kriptografi ditandai oleh diversifikasi algoritma dan aplikasi yang luas dalam berbagai aspek kehidupan digital. Pengembangan Advanced Encryption Standard (AES) pada tahun 2001 sebagai pengganti DES, evolusi elliptic curve cryptography untuk efisiensi yang lebih baik, dan emerging areas seperti quantum cryptography dan post-quantum cryptography menunjukkan bahwa kriptografi terus berkembang untuk menghadapi tantangan teknologi masa depan. Saat ini, kriptografi tidak hanya digunakan untuk komunikasi rahasia tetapi juga untuk digital signatures, authentication, blockchain technology, dan berbagai aplikasi keamanan siber yang telah menjadi infrastruktur critical dalam ekonomi digital global(Ketha, 2023).

Kriptografi modern memiliki empat tujuan fundamental yang dikenal sebagai security goals atau security objectives. Tujuan-tujuan ini membentuk foundation theoretical dan practical dari semua sistem kriptografi yang dikembangkan saat ini. Pemahaman yang mendalam tentang keempat tujuan ini

essential untuk merancang dan mengimplementasikan sistem keamanan yang efektif dan comprehensive(Maqsood et al., 2017).

1. Kerahasiaan (Confidentiality) merupakan prinsip yang memastikan bahwa hanya pengirim dan penerima yang berhak yang dapat mengakses dan memproses isi pesan, sehingga informasi yang dikirim tetap terlindungi dari pihak ketiga yang tidak berwenang. Untuk menjaga keamanan data, salah satu metode yang umum digunakan adalah teknik kriptografi.
2. Integrity (Integritas) memastikan bahwa data tidak mengalami perubahan, modifikasi, atau corruption selama proses penyimpanan atau transmisi. Integrity protection sangat penting dalam era digital di mana data dapat dengan mudah dimanipulasi tanpa meninggalkan jejak fisik. Cryptographic hash functions, message authentication codes (MAC), dan digital signatures merupakan tools utama yang digunakan untuk mencapai integrity. Violation terhadap integrity dapat mengakibatkan konsekuensi yang catastrophic, seperti dalam kasus medical records yang diubah atau financial transactions yang dimanipulasi. Modern integrity protection mechanisms tidak hanya mendeteksi unauthorized changes tetapi juga dapat mengidentifikasi specific parts dari data yang telah dimodifikasi.
3. Authentication (Otentikasi) verifies identity dari communicating parties dan memastikan bahwa pesan benar-benar berasal dari claimed sender. Authentication mencakup dua aspek utama: entity authentication yang verifies identity dari user atau system, dan data origin authentication yang confirms bahwa data berasal dari legitimate source. Digital signatures, public key certificates, dan challenge-response protocols merupakan mechanisms yang commonly digunakan untuk achieving authentication. Dalam network communications, authentication protocols seperti Kerberos dan public key infrastructures (PKI) provide frameworks untuk establishing trusted identities dalam distributed systems.
4. Non-repudiation (Non-penyangkalan) memastikan bahwa sender tidak dapat menyangkal telah mengirim pesan tertentu, dan receiver tidak dapat menyangkal telah menerima pesan tersebut. Non-repudiation particularly

important dalam legal dan business contexts di mana proof of communication or transaction diperlukan untuk dispute resolution. Digital signatures with trusted timestamping services menyediakan cryptographic evidence yang dapat digunakan dalam legal proceedings. Implementation non-repudiation memerlukan careful design untuk memastikan bahwa cryptographic evidence dapat withstand legal scrutiny dan technical challenges.

2.2 Data Digital

Data digital adalah segala bentuk informasi yang telah diubah ke dalam format digital (biner) sehingga dapat diproses, disimpan, dan dikirimkan melalui perangkat elektronik. Data ini bisa berupa teks, gambar, suara, video, atau dokumen digital lainnya yang membutuhkan perlindungan dari akses tidak sah selama proses transmisi atau penyimpanan di lingkungan digital. Dalam kriptografi, data digital menjadi objek utama yang diamankan melalui proses enkripsi, yaitu mengubah data asli (plaintext) menjadi bentuk terenkripsi (ciphertext) yang tidak dapat dipahami tanpa kunci dekripsi yang sesuai. Tujuan utama dari perlakuan ini adalah menjaga kerahasiaan, integritas, dan autentikasi data digital agar hanya pihak yang berwenang yang dapat mengakses atau memodifikasinya(Singh & Student, 2023)

2.3 Istilah umum dalam kriptigrafi

Dalam bidang ilmu kriptografi terdapat berbagai macam istilah-istilah yang digunakan sebagai foundation vocabulary untuk memahami concepts, processes, dan components yang terlibat dalam cryptographic systems. Pemahaman yang mendalam terhadap terminology ini essential untuk communication yang efektif dalam cryptographic community dan untuk proper implementation dari cryptographic solutions. Istilah-istilah ini telah berkembang sejalan dengan evolution kriptografi dari classical methods hingga modern computational approaches(Sarumaha et al., 2024). istilah-istilah yang digunakan tersebut adalah:

a. Pesan, plaintext dan chipertext

Pesan adalah informasi atau data yang ingin dikirimkan oleh pengirim kepada penerima. Jika pesan tersebut masih dalam bentuk aslinya yang dapat dibaca atau dipahami tanpa perlindungan khusus, maka disebut sebagai plaintext. Plaintext bisa berupa teks biasa, angka, atau data lainnya. Setelah pesan diubah melalui proses enkripsi menggunakan algoritma dan kunci tertentu, pesan tersebut menjadi ciphertext, yaitu bentuk terenkripsi yang tidak dapat dibaca atau dipahami tanpa proses dekripsi yang sesuai. Ciphertext digunakan untuk melindungi isi pesan dari pihak yang tidak berwenang.

b. Pengirim dan Penerima

Dalam sistem kriptografi, pengirim (sender) adalah pihak yang menginisiasi komunikasi dengan mengirimkan pesan. Pengirim bertanggung jawab untuk mengenkripsi pesan sebelum dikirim. Sementara itu, penerima (receiver) adalah pihak yang dituju untuk menerima pesan tersebut. Penerima akan melakukan proses dekripsi untuk mengubah ciphertext kembali menjadi plaintext yang dapat dipahami. Komunikasi yang aman membutuhkan kepercayaan dan kesepakatan antara pengirim dan penerima, khususnya dalam hal penggunaan algoritma dan pertukaran kunci.

c. Enkripsi dan dekripsi

Enkripsi adalah proses mengubah plaintext menjadi ciphertext dengan menggunakan algoritma dan kunci tertentu untuk menjaga kerahasiaan data. Tujuan utama enkripsi adalah memastikan bahwa pesan tidak dapat dibaca oleh pihak yang tidak memiliki otorisasi. Sebaliknya, dekripsi adalah proses mengubah ciphertext kembali ke bentuk plaintext menggunakan kunci yang sesuai. Dekripsi hanya dapat dilakukan oleh penerima yang sah yang memiliki akses ke kunci yang benar. Keduanya merupakan proses inti dalam sistem kriptografi.

d. Cipher dan key

Cipher adalah algoritma atau prosedur matematika yang digunakan untuk melakukan enkripsi dan dekripsi. Cipher menentukan bagaimana karakter atau bit dalam pesan diacak atau dimodifikasi untuk menghasilkan

ciphertext. Ada dua jenis cipher utama: cipher simetris (menggunakan satu kunci untuk enkripsi dan dekripsi) dan cipher asimetris (menggunakan pasangan kunci publik dan privat).
Key (kunci) adalah nilai atau informasi rahasia yang digunakan bersama dengan cipher untuk mengubah plaintext menjadi ciphertext, dan sebaliknya. Kekuatan sistem kriptografi sangat bergantung pada kerahasiaan dan kompleksitas kunci yang digunakan.

e. Kriptanalisis dan kriptologi
Kriptanalisis adalah ilmu dan teknik untuk menganalisis dan memecahkan sistem kriptografi tanpa memiliki akses ke kunci rahasia. Tujuan kriptanalisis adalah mencari celah atau kelemahan dalam algoritma atau implementasi kriptografi, yang dapat digunakan untuk membaca pesan terenkripsi secara tidak sah.
Kriptologi adalah cabang ilmu yang mencakup dua aspek utama: kriptografi (ilmu membuat sistem penyandian) dan kriptanalisis (ilmu memecahkan sistem penyandian). Dengan kata lain, kriptologi adalah ilmu yang mempelajari segala hal tentang penyandian dan pemecahan sandiri.

2.4 kriptografi kunci simetris dan asimetris

Kriptografi merupakan salah satu pilar utama dalam pengamanan informasi digital di era modern. Dengan semakin berkembangnya teknologi informasi, ancaman terhadap data digital pun turut meningkat, sehingga kebutuhan akan sistem keamanan yang kuat menjadi semakin mendesak. Salah satu aspek fundamental dalam kriptografi adalah penggunaan kunci sebagai alat untuk mengamankan data. Berdasarkan cara penggunaan kunci dalam proses enkripsi dan dekripsi, kriptografi dibagi menjadi dua jenis utama, yaitu kriptografi kunci simetris dan kriptografi kunci asimetris(Sarkar et al., 2024)**.**

Kedua pendekatan ini memiliki peran yang sangat penting dan saling melengkapi dalam sistem keamanan modern. Kriptografi kunci simetris dikenal dengan efisiensinya dalam pemrosesan data yang besar, sedangkan kriptografi kunci asimetris unggul dalam pengelolaan distribusi kunci dan autentikasi digital.

Pemahaman mendalam terhadap kedua jenis kriptografi ini sangat penting untuk merancang sistem keamanan data yang optimal dan sesuai dengan kebutuhan spesifik suatu lingkungan atau sistem.

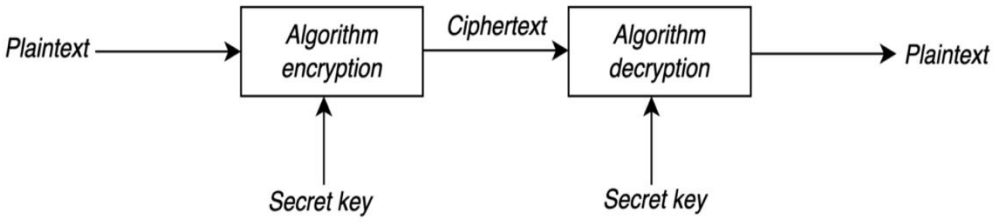
2.4.1 Algoritma Simetris

Kriptografi kunci simetris adalah bentuk kriptografi di mana proses enkripsi dan dekripsi dilakukan menggunakan satu kunci yang sama**.** Artinya, baik pengirim maupun penerima pesan harus memiliki kunci yang identik dan dijaga kerahasiaannya. Istilah "simetris" mengacu pada simetri penggunaan kunci tersebut dalam kedua proses tersebut. Kriptografi jenis ini telah digunakan sejak zaman kuno, dan berkembang pesat dengan ditemukannya algoritma-algoritma modern seperti DES (Data Encryption Standard)**,** AES (Advanced Encryption Standard)**,** dan Blowfish(Shallal et al., 2016)**.**

Sistem kriptografi simetris sangat bergantung pada keamanan kunci. Jika kunci tersebut bocor atau jatuh ke tangan yang salah, maka pesan yang dienkripsi akan dengan mudah didekripsi oleh pihak yang tidak berwenang. Oleh karena itu, tantangan utama dalam algoritma simetris bukan pada kekuatan algoritmanya, melainkan pada distribusi kunci yang aman(Thilagavathy et al., 2014)**.**

a. Proses Simetris

Dalam proses kriptografi simetris, pengirim dan penerima terlebih dahulu harus sepakat menggunakan satu kunci rahasia yang sama. Setelah itu, pengirim akan mengenkripsi pesan asli (plaintext) menggunakan kunci tersebut untuk menghasilkan ciphertext. Ciphertext kemudian dikirimkan kepada penerima. Setelah diterima, penerima menggunakan kunci yang sama untuk mendekripsi ciphertext dan mengubahnya kembali menjadi plaintext.



Gambar 2. 1 Alur kerja algoritma kriptografi simetris

proses algoritma simetris melibatkan tahapan-tahapan berikut:

1. Pengirim memasukkan data asli (plaintext).
2. Plaintext tersebut kemudian dienkripsi menggunakan kunci simetris yang sudah dibuat sebelumnya.
3. Hasil enkripsi berupa ciphertext (bersama dengan kuncinya) akan dikirimkan kepada penerima.
4. Penerima akan mendekripsi ciphertext tersebut menggunakan kunci yang sama yang diterima dari pengirim.
5. Proses dekripsi ini akan menghasilkan kembali data asli (plaintext).

Jenis dan contoh algoritma Simetris Terdapat dua kategori utama dalam algoritma kriptografi simetris, yaitu block cipher dan stream cipher. Beberapa contoh algoritma simetris yang dikenal antara lain Spritz, Rabbit Stream, RC4, TwoFish, dan Rijndae

Kelebihan Algoritma Simetris

1. Cepat dan efisien: Proses enkripsi dan dekripsi pada algoritma simetris umumnya jauh lebih cepat dibandingkan algoritma asimetris, sehingga cocok digunakan untuk mengamankan data dalam jumlah besar.
2. Lebih hemat sumber daya: Karena kompleksitas algoritma yang lebih rendah, algoritma simetris tidak membutuhkan daya komputasi tinggi, sehingga

sangat cocok untuk digunakan pada perangkat dengan sumber daya terbatas seperti IoT dan embedded system.

3. Implementasi sederhana: Struktur algoritmanya mudah dipahami dan diterapkan, menjadikannya pilihan ideal dalam banyak aplikasi praktis.

Kelemahan Algoritma Simetris

1. Distribusi kunci yang sulit: Tantangan utama dalam algoritma simetris adalah bagaimana cara mengirimkan atau membagikan kunci rahasia kepada pihak penerima tanpa terekspos kepada pihak ketiga.
2. Risiko kebocoran kunci: Jika kunci diketahui pihak yang tidak berwenang, maka seluruh sistem keamanan menjadi tidak berguna, karena semua pesan dapat dibaca.
3. Kurang fleksibel untuk komunikasi multipihak: Dalam sistem yang melibatkan banyak pengguna, dibutuhkan banyak kunci berbeda, yang menambah kompleksitas pengelolaan.

2.4.2 Algoritma Asimetris

Berbeda dengan pendekatan simetris, kriptografi kunci asimetris menggunakan sepasang kunci yang berbeda, yaitu kunci publik (public key) dan kunci privat (private key)**.** Kunci publik dapat dibagikan secara luas kepada siapa saja yang ingin mengirimkan pesan, sedangkan kunci privat harus disimpan secara rahasia oleh penerima. Konsep utama dari algoritma ini adalah bahwa pesan yang dienkripsi dengan kunci publik hanya bisa didekripsi oleh kunci privat yang sesuai, dan sebaliknya. Algoritma ini diperkenalkan untuk mengatasi masalah distribusi kunci yang menjadi kelemahan utama dari sistem simetris(Muneer Bani Yassein, 2017).

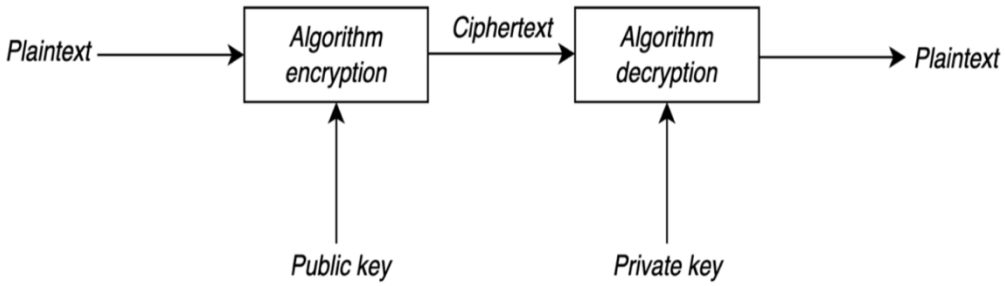
Kriptografi asimetris merupakan landasan utama dari sistem keamanan dalam komunikasi elektronik modern seperti SSL/TLS**,** sertifikat digital**,** tanda tangan digital**,** serta sistem transaksi dalam **e-**commerce. Algoritma yang paling

populer dalam kategori ini adalah RSA (Rivest-Shamir-Adleman)**,** ElGamal**,** ECC (Elliptic Curve Cryptography)**,** dan Cramer-Shoup(Al-Shabi, 2019)**.**

b. Proses Asimetris

Dalam proses asimetris, pengguna akan menghasilkan sepasang kunci: satu kunci publik dan satu kunci privat. Pengirim mengenkripsi pesan menggunakan kunci publik milik penerima. Karena hanya kunci privat yang dapat mendekripsi pesan tersebut, maka hanya penerima sah yang dapat membaca isinya.

Sebaliknya, dalam konteks tanda tangan digital, pengirim dapat mengenkripsi hash dari pesan menggunakan kunci privat, dan penerima dapat memverifikasi keaslian tanda tangan tersebut dengan kunci publik. Ini menjamin bahwa pesan benar-benar berasal dari pengirim yang sah.



Gambar 2. 2 Alur kerja algoritma kriptografi asimetris

Kelebihan Algoritma Asimetris

1. Distribusi kunci lebih mudah dan aman: Tidak perlu mendistribusikan kunci privat, hanya kunci publik yang dibagikan. Hal ini mengurangi risiko kebocoran kunci.
2. Mendukung autentikasi dan integritas data: Dengan dukungan terhadap tanda tangan digital, algoritma ini menjamin keaslian pengirim dan keutuhan pesan.

3. Ideal untuk sistem komunikasi terbuka: Cocok digunakan dalam sistem komunikasi antar pihak yang tidak saling mengenal sebelumnya, seperti dalam sistem internet dan e-commerce.

Kelemahan Algoritma Asimetris

1. Lebih lambat dibandingkan simetris: Karena algoritma asimetris menggunakan operasi matematika yang lebih kompleks, proses enkripsi dan dekripsi membutuhkan waktu lebih lama.
2. Membutuhkan sumber daya komputasi tinggi: Tidak cocok untuk perangkat dengan kapasitas terbatas atau untuk mengenkripsi data dalam jumlah besar secara langsung.
3. Tidak efisien untuk data besar: Dalam praktiknya, algoritma asimetris sering digunakan hanya untuk mengenkripsi kunci sesi, bukan seluruh isi pesan, karena alasan performa.

2.5 Kombinasi

Dalam dunia kriptografi modern, kebutuhan akan sistem keamanan yang kuat dan efisien mendorong lahirnya berbagai pendekatan baru, salah satunya adalah pendekatan kriptografi hybrid. Hybrid merupakan kombinasi dari dua jenis algoritma kriptografi, yaitu algoritma simetris dan algoritma asimetris, yang digabungkan untuk memanfaatkan keunggulan dari masing-masing jenis kriptografi dan mengatasi kelemahan yang dimiliki jika digunakan secara tunggal(Sarumaha et al., 2023).

Kelebihan Kriptografi Hybrid

1. Menggabungkan keamanan dan efisiensi: Pendekatan hybrid memanfaatkan keamanan tinggi dari algoritma asimetris untuk pengiriman kunci dan efisiensi dari algoritma simetris untuk pengolahan data, sehingga menghasilkan sistem yang seimbang.
2. Skalabilitas lebih baik: Cocok digunakan untuk sistem besar dan kompleks, seperti sistem transaksi online, layanan cloud, dan aplikasi mobile.

3. Meningkatkan kecepatan enkripsi: Karena data utama dienkripsi menggunakan algoritma simetris yang cepat, hybrid tetap menawarkan performa tinggi meskipun menggunakan enkripsi ganda.
4. Perlindungan terhadap serangan: Hybrid membantu melindungi dari berbagai jenis serangan, seperti serangan brute-force terhadap kunci simetris atau serangan penyadapan terhadap jalur distribusi kunci.

2.6 Landasan Matematika Kriptografi

Kriptografi modern tidak hanya bergantung pada algoritma pemrograman dan sistem digital, tetapi juga sangat bergantung pada prinsip-prinsip matematika. Banyak algoritma kriptografi, terutama yang berbasis pada kriptografi kunci publik**,** menggunakan konsep matematika dalam pengolahan data, terutama untuk menghasilkan kunci, melakukan proses enkripsi, dan juga dekripsi. Landasan matematika ini memberikan kekuatan keamanan terhadap algoritma yang digunakan, karena masalah-masalah matematis tertentu memerlukan waktu komputasi yang sangat besar untuk diselesaikan tanpa kunci rahasia.
Berikut adalah beberapa landasan matematika penting yang digunakan dalam kriptografi:

2.6.1 Bilangan Prima

Bilangan prima adalah bilangan asli yang hanya memiliki dua faktor, yaitu 1 dan dirinya sendiri. Artinya, bilangan prima tidak dapat dibagi secara habis oleh bilangan lain selain 1 dan bilangan itu sendiri. Bilangan prima merupakan elemen penting dalam kriptografi karena digunakan dalam algoritma kriptografi asimetris seperti RSA, termasuk dalam pembangkitan kunci dan proses dekomposisi faktorisasi. Contoh bilangan prima : 2,3,5,7,11 dan lain sebagainya(Aribowo, n.d.).

2.6.2 Greatest Common Divisor (GCD)

GCD atau *Greatest Common Divisor* adalah faktor terbesar yang sama-sama membagi habis dua bilangan atau lebih. Dalam konteks kriptografi, GCD digunakan untuk memastikan dua bilangan relatif prima (tidak memiliki faktor

pembagi yang sama selain 1), misalnya saat membangkitkan kunci publik dan privat dalam RSA(Khoiruddin et al., 2016).

Contoh Berapa GCD dari 36 dan 11 ?

36 mod 11 = 3

11 mod 3 = 2

3 mod 2 = 1

1 mod 1 = 0

Jadi GCD dari 36 dan 11 adalah 1

2.6.3 Relatif Prima

Dua bilangan dikatakan relatif prima jika GCD-nya adalah 1, artinya mereka tidak memiliki faktor pembagi yang sama kecuali angka 1. Dalam kriptografi, bilangan yang relatif prima sering digunakan dalam algoritma RSA untuk memilih nilai eksponen publik dan fungsi totien Euler.

Conoth : Apakah 30 dan 7 relatif prima ?

30 mod 7 = 2

7 mod 2 = 1

2 mod 1 = 0

Jadi 30 dan 7 adalah relative prima

2.6.4 Exclusive OR (XOR)

Operasi Exclusive OR (XOR) merupakan salah satu operator logika dasar dalam sistem biner yang banyak digunakan dalam bidang kriptografi. XOR adalah operasi Boolean yang menghasilkan nilai benar (1) apabila kedua operand memiliki nilai yang berbeda, dan menghasilkan nilai salah (0) apabila kedua operand memiliki nilai yang sama. Dalam representasi biner, operasi ini sering dinyatakan sebagai penjumlahan modulo dua. Salah satu karakteristik penting dari XOR adalah sifatnya yang *reversibel* atau *self-inverse*, di mana jika suatu nilai dienkripsi menggunakan XOR dengan sebuah kunci, maka proses dekripsi dapat dilakukan dengan operasi XOR yang sama menggunakan kunci tersebut. Sifat ini menjadikan

XOR sangat efisien dan banyak diterapkan dalam berbagai algoritma kriptografi, terutama pada skema enkripsi sederhana seperti One-Time Pad. Dalam konteks tersebut, XOR digunakan untuk menggabungkan plaintext dengan kunci acak sehingga menghasilkan ciphertext yang sulit dianalisis tanpa mengetahui kunci yang digunakan. Oleh karena itu, XOR memiliki peranan penting sebagai dasar dalam mekanisme enkripsi modern, baik sebagai komponen utama maupun sebagai bagian dari operasi kriptografi yang lebih kompleks.

Contoh:

U= 01010101

N= 01001110

Berapa hasil XOR dari U dan N?

01010101
01001110 XOR
00011011

2.7 Pembangkitan Bilangan Prima

Dalam kriptografi, bilangan prima memiliki peranan yang sangat penting, terutama dalam sistem kriptografi kunci publik seperti RSA dan algoritma Cramer-Shoup. Bilangan prima digunakan sebagai basis pembentukan kunci publik dan privat karena sifat matematisnya yang sulit untuk difaktorkan kembali menjadi bilangan asal, sehingga memberikan tingkat keamanan yang tinggi.

Pembangkitan bilangan prima adalah proses untuk menghasilkan bilangan yang hanya memiliki dua faktor, yaitu 1 dan dirinya sendiri. Dalam praktiknya, terutama pada skala besar (ratusan atau ribuan bit), pembangkitan bilangan prima dilakukan menggunakan metode acak (probabilistik) dan kemudian diuji keprimaannya menggunakan algoritma pengujian bilangan prima seperti uji Fermat**.** Teorema Kecil Fermat merupakan salah satu teori dasar dalam teori bilangan yang sering digunakan dalam pengujian bilangan prima secara probabilistik(Samandari et al., 2023).

Rumus Fermat

$a^{p-1} \ mod \ p \ = 1$ pasti bilangan prima

a = Pilih secara acak bilangan bulat positif dimana $1 < a < p$.

Contoh : apakah 7 bilangan prima?

P = 7

$= 3^{7-1} \ mod \ 7$

$= 3^{7-1} \ mod \ 7$

= 1

Jadi 7 adalah bilangan prima

2.8 Know Plaintext Attack (KPA)

Known-Plaintext Attack (KPA) adalah jenis serangan kriptografi di mana penyerang memiliki akses sekaligus terhadap sebagian pasangan *plaintext* (pesan asli) dan *ciphertext* (pesan terenkripsi) yang sesuai. Dengan informasi ini, penyerang berusaha mengekstrak kunci enkripsi atau menemukan pola dalam algoritma sehingga dapat mendekripsi pesan lain yang tidak diketahui. Intinya, KPA memanfaatkan fakta bahwa penyerang sudah "tahu sebagian isi pesan" untuk membongkar keseluruhan sistem enkripsi(Taqwim et al., 2021).

Rumus KPA untuk OTP

$K=(C \oplus P)$

Keterangan :

K = Kunci

C = Ciphertext

P = Plaintext

$\oplus$ = XOR

2.9 Algoritma OTP

One-Time Pad (OTP) adalah salah satu algoritma kriptografi simetris yang paling sederhana namun sangat kuat secara teori. OTP menggunakan kunci yang acak dan hanya digunakan satu kali untuk mengenkripsi dan mendekripsi pesan. Kunci OTP harus sepanjang pesan dan benar-benar acak, serta tidak boleh

digunakan kembali, karena jika tidak, keamanan pesan dapat sangat mudah dikompromikan(Lestari et al., 2021).

2.9.1 Pembangkitan Kunci Algoritma OTP

Langkah-langkah pembangkitan kunci pada OTP:

1. Tentukan panjang pesan: Hitung jumlah karakter dari pesan asli (plaintext).
2. Buat kunci acak yang memiliki panjang yang sama dengan pesan.

2.9.2 Metode Enkripsi

Rumus enkripsi OTP :

$Ci=Pi \oplus Ki$

Keterangan :

Ci = nilai decimal karakter *chipertext*

Pi = nilai decimal karakter *plaintext*

Ki = nilai decimal karakter kunci

$\oplus$ = XOR

2.9.3 Metode Dekripsi

Dekripsi dilakukan dengan mengurangkan nilai karakter ciphertext dengan nilai karakter dari kunci, lalu di-*modulo* 26 untuk mendapatkan plaintext kembali.

$Pi=Ci \oplus Ki$

Keterangan :

Ci = nilai decimal karakter *chipertext*

Pi = nilai decimal karakter *plaintext*

Ki = nilai decimal karakter kunci

Contoh perhitungan manual dari algoritma OTP

Plaintext = gilang

Key(kunnci) = arka

a. Ubah plainteks dan kunci kedalam decimal

Tabel 2. 1 Convert Plainnteks dan kunci ke decimal

Plainteks	g	i	l	a	n	g
Plainteks (*Desimal ASCII*)	103	105	108	97	110	103
Key	a	r	k	a	a	r
Key (*Decimal*)	97	114	107	97	97	114

b. Proses enkripsi/menyembunyikan pesan

Tabel 2. 2 Proses enkripsi/menyembunyikan pesan OTP

Plainteks(*Desimal*)	103	105	108	97	110	103
Key (*Decimal*)	97	114	107	97	97	114
Ubah seluruh plaintek dan key ke dalam bentuk biner agar dapat di XOR						

- Ci=(Pi⊕Ki)

  Ci = $\frac{01100111}{01100001}$ XOR

  Ci = 00000110 = 6

- Ci=(Pi⊕Ki)

  Ci = $\frac{01101001}{01110010}$ XOR

Ci = 00011011 = 27

- Ci=(Pi⊕Ki)

  Ci = $\frac{01101100}{01101011}$ XOR

  Ci = 00000111= 7

- Ci=(Pi⊕Ki)

  Ci = $\frac{01100001}{01100001}$ XOR

  Ci = 00000000= 0

- Ci=(Pi⊕Ki)

  Ci = $\frac{01101110}{01100001}$ XOR

  Ci = 00001111= 15

- Ci=(Pi⊕Ki)

  Ci = $\frac{01100111}{01110010}$ XOR

  Ci = 00010101= 21

Tabel 2. 3 Hasil Enkripsi Chipertext

Plainteks asli (*Desimal*)	103	105	108	97	110	103
Key (*Decimal*)	97	114	107	97	97	114
Chipertex	6	27	7	0	15	21

Jadi yang dikrim ke penerima adalah *chipertext* dan *key*

c. Dekripsi / mengembalikan pesan yang terkunci

key dan *plaintext* yang diterima dari *sender* :

Tabel 2. 4 Dekripsi / mengembalikan pesan yang terkunci OTP

Chipertex	6	27	7	0	15	21
Key (*Decimal*)	97	114	107	97	97	114
Ubah seluruh plaintek dan key ke dalam bentuk biner agar dapat di XOR						

- Pi=(Ci⊕Ki)

  Pi = $\frac{00000110}{01100001}$ XOR

  Pi = 01100111= 103 = g

- Pi=(Ci⊕Ki)

  Pi = $\frac{00011011}{01110010}$ XOR

  Pi = 01101001= 105 = i

- Pi=(Ci⊕Ki)

  Pi = $\frac{00000111}{01101011}$ XOR

  Pi = 01101100= 108 = l

- Pi=(Pi⊕Ki)

  Pi = $\frac{00000000}{01100001}$ XOR

  Pi = 01100001= 97 = a

- Pi=(Ci⊕Ki)

Pi = $\frac{00001111}{01100001}$ XOR

Pi = 01101110= n

- Pi=(Ci⊕Ki)

Pi = $\frac{00010101}{01110010}$ XOR

Pi = 01100111 = 103 = g

Tabel 2. 5 Hasil convert decimal ke plaintext semula

Plaintext semula / Decimal	103	105	108	97	110	103
Plaintext semula / ASCII	g	i	l	a	n	g

Jadi proses enkripsi dan dekripsi berhasil dilakukan dengan menggunakan algoritma OTP.

Kelemahan dari OTP

Secara teori OTP tak dapat dipecahkan bila kuncinya acak sempurna, sekali pakai, sepanjang pesan, dan dirahasiakan.

1. Reuse kunci (*two-time pad*) atau menggunakan kunci yang sama untuk dua pesan yang di enkripsi
2. Apabila kunci tidak benar-benar acak, karena jika plainteks yang diinput terlalu Panjang dan kunci tidak sepanjang plainteks maka kunci akan diualang

   Contoh :

   Plainteks = Potensi Utama

   Key = gilang

   Maka kunci akan berulang sepanajang plainteks

Key berulang = gilanggilangg

3. Distribusi & penyimpanan kunci sangat sulit (harus sepanjang pesan dan benar-benar rahasia sehinggat membutuhkan biaya besar untuk mengirim kunnci).

Know Plaintext Attack

1. Penyerang mendapatkan chipertext yang tersandi yaitu [6,27,7,0,15,21] yang di dapat dari tempat pengiriman misalnya internet, wa (Misalnya penyerang menyadap WA dan internet)
2. Penyerang mengetahui plainteks awal mislnya [g,i,l] di decimal [103,105,108]
3. KPA akan pulihkan kunci dari bagian yang di ketahui

Rumus KPA

$K=(C \oplus P)$

$K1 = 6 \oplus 103 = 97 = a$

$K2 = 27 \oplus 105 = 114 = r$

$K3 = 7 \oplus 108 = 107 = k$

Lalu bagaimana KPA mendapatkan plainteks sepenuhya

Karena system menggunakan kunci berulang maka di uji pola dengan melihat apakah hasil dekripsi masuk akal (huruf biasa)

a. Uji kunci yang didapat ‘ark’
    - Perkirakan K4 = a (mengulang dari awal)
    - $P4 = 0 \oplus 97 = 97 = a$ (masuk akal)
    - Perkirakan K5 = r
    - $P5 = 15 \oplus 114 = 125 = \}$ (tidak masuk akal) maka dilakuka revisi keystream
b. Revisi keystream
    - Coba K5 = a (ada dua ‘a’ berturut)
    - $P5 = 15 \oplus 97 = 110 = n$ (masuk akal) lanjut posisi 6
    - K6 = r

- P6 = 21 ⊕ 114 = 103 = g (wajar)

Jadi keystream 6 byte yang konsisten adalah “arkaar”

Berdasarkan kelemahan dari OTP yang menggunakan kunci yang sama dalam enkripsi pesan yang berbeda, maka akan mudah ditebak plainteksnya dan dapat dilihat pada perhitungan manual di bawah ini :

Plainteks = POTENSI UTAMA
Menggunakan key yang sama untu enkripsi POTENSI UTAMA
Hitung Chipertext = P ⊕ K
Hasilnya = [49, 61, 63, 36, 47, 33, 34, 65, 52, 38, 42, 44, 32]
Karena penyerang sudah menegetahui kunci yang digunakan sebelumnya di gilang maka dengan mudah mengembalikan pesan:

Tabel 2. 6 Pengujian menggunakan KPA

Ciphertext (Decimal)	Key berulang	P = C ⊕ K Decimal	Plainteks ASCII
49	a = 97	80	P
61	r =114	79	O
63	k =107	84	T
36	a = 97	69	E
47	a = 97	78	N
33	r =114	83	S
34	k =107	73	I
65	a = 97	32	“SPASI”
52	a = 97	85	U
38	r =114	84	T
42	k =107	65	A
44	a = 97	77	M
32	a = 97	65	A

Jadi dengan metode Know Planteks Attack ini akan dengan mudah mendapatkan pesan jika menggunakanan kunci berulang dan mengguakan kunci yang sama dalam mengenkripsi pesan yang berbeda.

2.7.3 Hasil Pengujian Sitem KPA

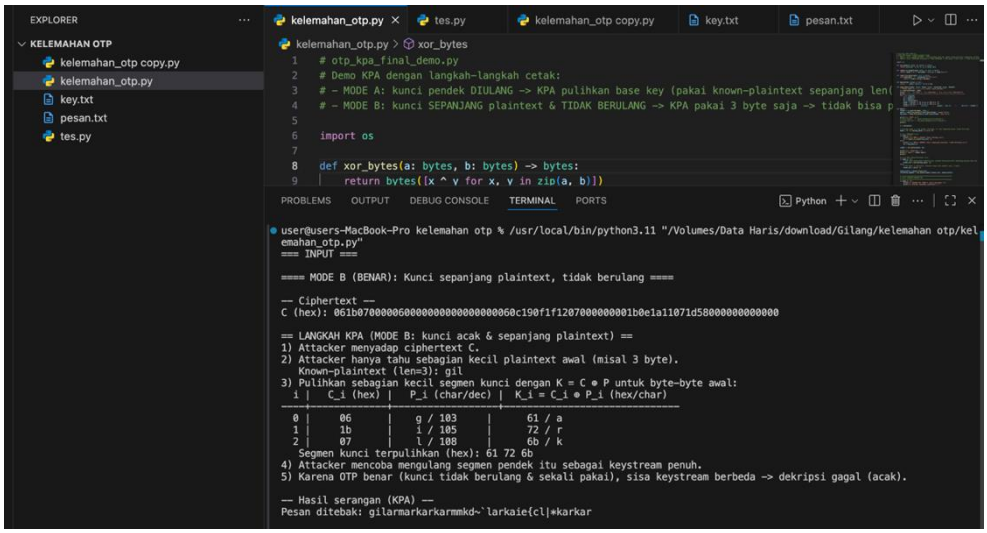
Hasil pengujian sistem dengan metode *Known-Plaintext Attack* (KPA) menunjukkan bahwa ketika sebagian pasangan plaintext dan ciphertext diketahui, sistem mampu mengidentifikasi kunci enkripsi yang digunakan dengan tingkat akurasi yang tinggi. Proses pengujian dilakukan dengan cara membandingkan pola pergeseran antara plaintext dan ciphertext, kemudian sistem secara otomatis menurunkan nilai kunci yang konsisten. Dari serangkaian percobaan, terlihat bahwa semakin panjang bagian plaintext yang diketahui, semakin cepat sistem menemukan kunci yang benar serta dapat mendekripsi keseluruhan ciphertext dengan tepat. Hal ini membuktikan bahwa implementasi algoritma serangan KPA berjalan efektif dan dapat digunakan untuk mengevaluasi kelemahan kunci OTP yang digunakan secara berulang

Pengujian 1

Plainteks : gilang sedang belajar di potensi utama

Key : arkana sedang diunusu di kantin1 utama

Hasil secara system menunjukan bahwa chipertext tidak dapat dipecahkan karena kunci sangat acak dan sepanajang plaintext, berikut hasil KPA “gilarmarkarkarmmkd~`larkaie{cl|*karkar”



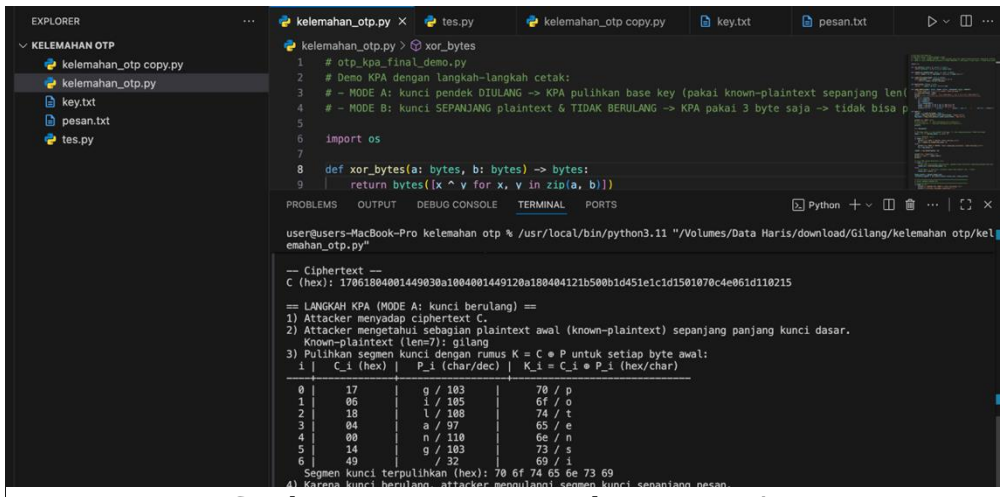
Gambar 2. 3 pengujian 1 pada sistem KPA

Pengujian 2

Plainteks : gilang sedang belajar di potensi utama

Key : potensi

Hasil secara system menunjukan ciphertext dan key dapat di pecah, karena key yang digunakan secara berulang dan tidak sepanjang plaintext, berikut hasil sitem “gilang sedang belajar di potensi utama”



Gambar 2. 4 pengujian 2 pada sistem KPA

2.10 Algoritma Cramer-shoup

Algoritma Cramer-Shoup adalah salah satu algoritma kriptografi kunci publik (asymmetric encryption) yang dikembangkan oleh Ronald Cramer dan Victor Shoup pada tahun 1998. Algoritma ini merupakan penyempurnaan dari algoritma ElGamal dan merupakan algoritma pertama yang terbukti aman terhadap serangan chosen ciphertext attack (CCA-secure) pada model kriptografi modern(General Chair, 2019).

Cramer-Shoup adalah metode kriptografi yang bekerja dengan memanfaatkan teori bilangan prima dan konsep matematika tertentu. Metode ini juga menggunakan cara pengacakan yang sudah ditentukan sebelumnya (disebut pengacakan deterministik) untuk membuat sistem keamanannya menjadi lebih kuat dan sulit ditembus.

2.10.1 Cara kerja algoritma cramer-shoup

1. *Generate*/Pembangkita kunci

a. Pada tahap pembuatan kunci, langkah-langkah yang dilakukan antara lain: menghasilkan nilai $g1$, $g2$, dan sebuah bilangan prima q.

b. Selanjutnya, dipilih enam bilangan acak yaitu $x1$, $x2$, $y1$, $y2$, dan z yang nilainya berada dalam rentang 0 hingga q-1.

c. Setelah itu, dilakukan perhitungan terhadap variabel c, d, dan h menggunakan rumus berikut:

$c = g1^{x1} * g2^{x2} \bmod q$

$d = g1^{y1} * g2^{y2} \bmod q$

$h = g1^{z} \bmod q$

sehingga didapatkan kunci public (g1,g2,c,d,h,H) dank unci *private* (x1,x2,y1,y2,z).

2. Enkripsi

1. Tentukan pesan dan konversi ke bilangann decimal sesuai table ASCII

2. Sebuah bilangan acak r dipilih dari rentang 0 hingga $q-1$. Setelah itu, nilai $u1$, $u2$, e, a, dan v dihitung dengan memanfaatkan kunci publik. Adapun rumus perhitungannya adalah sebagai berikut:

   $u1 = g1^r \bmod q$

   $u2 = g2^r \bmod q$

   $e = h^r * m \bmod q$

   $a = u1 + u1 \bmod e$

3. Dekripsi

Kembalikan pesan tersandi dengan menggunkan rumus:

$m = e\,(u1^{-1})^z \bmod q$

Contoh perhitungan manual dengan algoritma cramer-shoup

1. Bangkitkan kunci
   - Pilih bilangan prima besar q = 251
   - Bangkitkan bilangan acak g1 = 4 dan g2 =6
   - Bangkitkan bilangan acak x1, x2, y1, y2, z

     X1 = 12

     X2 = 10

     Y1 = 14

     Y2 = 8

     Z = 18
   - Hitung nilai

     $c = g1^{x1} * g2^{x2} \bmod q$

     $c = 4^{12} * 6^{10} \bmod 251$

     c = 113

     $d = g1^{xy1} * g2^{y2} \bmod q$

     $d = 4^{14} * 6^{8} \bmod 251$

     d = 106

$h = g1^z \bmod q$

$h = 4^{18} \bmod 251$

$h = 211$

jadi *private key* (x1=12, x2 = 10, y1=14, y2 = 8, z = 18)

publick key (g1 = 4, g2 = 6, c = 113, d=106, h =211)

2. Proses enkripsi

- Pesan m = G = 71 (Karakter G di uabah dalam decimal kode ASCII)
- Pilih bilangan acak r dalam rentang 0 sampai q1 kemudian hitung nilai u1,u2,e dan a

$r = 11$

$u1 = g1^r \bmod q$

$u1 = 4^{11} \bmod 251$

$u1 = 94$

$u2 = g2^r \bmod q$

$u2 = 6^{11} \bmod 251$

$u2 = 150$

$e = h^r * m \bmod q$

$e = 211^{11} * 71 \bmod 251$

$e = 177$

$a = u1 + u2 \bmod e$

$a = 94 + 150 \bmod 177$

$a = 244$

Jadi chipertext (u1 = 94, u2 =150, e = 177)

3. Proses dekripsi

$m = e\ (u1^{-1})^z \bmod q$

$m = 177\ (94^{-1})^{18} \bmod 251$

$m = 177\ (243\ )^{18} \bmod 251$

$m = 71$

jadi proses enkripsi dan dekripsi dengan menggunakan algoritma cramer-shoup berhasil dilakukan.

2.11 Penelitian Terkait

No	Judul Peneliian	Tahun	Hasil Penelitan
1	Hybrid Cryptographic Technique Using OTP:RSA	Ouk et al., 2020	Penggunaan kombinasi algoritma RSA dan OTP—yang dikenal sebagai algoritma hibrida merupakan pendekatan yang tepat untuk meningkatkan keamanan kriptografi. Algoritma hibrida ini telah diimplementasikan menggunakan Python 3. Algoritma hibrida yang diusulkan terbukti dapat mengamankan data, serta mampu menjaga integritas data dan koneksi jaringan karena proses enkripsi ganda yang dilakukan oleh RSA dan OTP saling melengkapi kekurangan masing-masing. Bahkan jika peretas berhasil membobol RSA, mereka tetap tidak dapat menembus lapisan enkripsi terakhir yang menggunakan OTP, sehingga sistem menjadi lebih aman dan tidak rentan terhadap berbagai ancaman(Ouk et al., 2020).
2	Pengamanan Data Ekstensi File Pdf Dengan	(Musa Asyari Hidayat	menyimpulkan bahwa proses enkripsi dan dekripsi file KTP menggunakan algoritma AES 128-bit dengan mode CBC untuk

	Algoritma Aes dan OTP	Jati, n.d.2021).	menjaga keamanan data. Verifikasi dilakukan melalui kode OTP yang dikirim ke email pengguna dan berlaku selama 300 detik. Hasil enkripsi berupa file PDF yang hanya dapat diakses oleh pihak berwenang setelah memasukkan OTP yang sesuai. Proses ini menjamin keamanan dan integritas dokumen selama pertukaran data. Disarankan penggunaan metode kunci asimetris dan peningkatan verifikasi dua langkah untuk memperkuat sistem keamanan
3	Sign-Then-Encrypt Scheme with Cramer-Shoup Cryptosystem and Dissanayake Digital Signature	Bahri et al., 2024)	menyimpulkan bahwa proses enkripsi dan dekripsi dengan algoritma Cramer-Shoup, diketahui bahwa panjang karakter memengaruhi waktu eksekusi secara linear. Dari hasil simulasi efek avalanche, tanda tangan digital Dissanayake memperoleh nilai rata-rata efek avalanche sebesar 50,14%, sedangkan algoritma Cramer-Shoup sebesar 50,29%. Implementasi skema sign-then-encrypt dapat menjaga keamanan dengan mengenkripsi data sekaligus menjamin keasliannya melalui penambahan tanda tangan digital
4	A Hybrid Cryptosystem using DNA, OTP and RSA	Begum et al., 2020	menyimpulkan bahwa teknik enkripsi teks berbasis kombinasi DNA, One Time Pad (OTP), dan RSA mampu memberikan tingkat keamanan tinggi melalui tiga

			lapisan enkripsi. Sistem ini merupakan pendekatan hibrida yang menggabungkan kriptografi simetris dan asimetris, sehingga lebih aman dibandingkan metode yang hanya menggunakan sistem simetris. Namun, penelitian ini juga mengidentifikasi beberapa keterbatasan, seperti ukuran ciphertext yang dibatasi kelipatan 8 bit saat mengenkripsi nilai ASCII dengan RSA, serta kebutuhan untuk berbagi kunci biner secara aman antara pengirim dan penerima. Kedua hal ini menjadi perhatian untuk pengembangan lebih lanjut di masa depan
5	Implementasi Algoritma Rsa Dan One Time Password (Otp) Untuk Pengamanan Data Pengguna Dan Proses Transaksi Pada Website E-Commerce	Calvin Christian, n.d. 2023	menyimpulkan bahwa Pengamanan dengan verifikasi One Time Password (OTP) berhasil dilakukan, karena pada saat pengujian sistem kode OTP berhasil terkirim ke email pengguna dan hasil pengujian verifikasi dengan empat kondisi berbeda sudah mendapatkan hasil yang sesuai, sehingga proses verifikasi OTP hanya akan berhasil dengan kondisi kode OTP yang dimasukkan benar dan masa berlaku kode OTP belum berakhir
6	Implementation of Hybrid Cryptosystem using Rabin-p Algorithm and	Budiman et al., 2021	menyimpulkan bahwa algoritma One Time Pad (OTP) efektif digunakan untuk mengamankan pesan citra, sedangkan algoritma Rabin-p digunakan untuk mengamankan kunci. Penelitian ini juga

	One Time Pad to Secure Images		menunjukkan bahwa semakin besar ukuran citra, maka waktu yang dibutuhkan dalam proses randomizing kunci OTP, enkripsi, dan dekripsi akan semakin lama, dengan proses dekripsi memakan waktu lebih panjang dibandingkan proses enkripsi. Selain itu, ukuran citra asli dan citra terenkripsi (cipherimage) tidak mengalami perubahan selama proses enkripsi dan dekripsi berlangsung
7	Data Security In Cloud Computing Using Cramer–Shoup Cryptosystem	Dr. S. Raju, 2022	menyimpulkan bahwa Cloud Computing merupakan paradigma baru yang masih berkembang, di mana komputasi dipandang sebagai layanan berdasarkan permintaan, namun mengakibatkan hilangnya kontrol atas data oleh organisasi yang menggunakannya. Oleh karena itu, tingkat perlindungan data harus disesuaikan dengan nilai data tersebut. Dalam penelitian ini, keamanan cloud bergantung pada komputasi tepercaya dan kriptografi, dan hanya pengguna yang berwenang yang dapat mengakses data. Bahkan jika pihak tidak sah memperoleh data tersebut, mereka akan kesulitan mendekripsi dan mengembalikan data asli. Untuk itu, kriptosistem Cramer–Shoup diterapkan sebagai solusi untuk menjamin keamanan data dalam lingkungan komputasi awan

8	Message Security Using One Time Pad and AES Hybrid Cryptography	Tun et al., 2020	menyimpulkan bahwa penerapan kriptografi hibrida yang menggabungkan algoritma One Time Pad (OTP) dan AES mampu meningkatkan keamanan pesan. Evaluasi kinerja berdasarkan waktu proses, penggunaan memori, dan efek avalanche menunjukkan bahwa algoritma OTP-AES mengurangi sedikit konsumsi waktu dan memori dibandingkan jika OTP dan AES dijalankan secara terpisah. Selain itu, algoritma OTP-AES menunjukkan peningkatan signifikan dalam efek avalanche dibandingkan AES murni, sehingga menghasilkan tingkat keamanan pesan yang lebih baik dengan menggabungkan kekuatan XOR dari OTP dan struktur enkripsi kuat dari AES
9	Tversky Indexive Cramer–Shoup Cryptography Based Deep Structured Belief Neural Learning for Secured Routing in Manet	Navamani & Elamathi, 2022	menyimpulkan bahwa dengan menerapkan kriptosistem Cramer–Shoup dapat mencegah akses dari node yang tidak sah. Hasil simulasi menunjukkan bahwa metode ini memberikan performa yang lebih baik dibandingkan metode routing konvensional, terutama dalam hal kerahasiaan data yang lebih tinggi, rasio pengiriman paket yang lebih baik, penurunan tingkat kehilangan paket, serta waktu tunda yang lebih rendah